A multiprecision C++ library for matrix-product-state simulation of quantum computing: Evaluation of numerical errors

Akira SaiToh

Quantum Information Science Theory Group, National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda, Tokyo 101-8430, Japan

E-mail: akirasaitoh@nii.ac.jp

Abstract. The time-dependent matrix-product-state (TDMPS) simulation method has been used for numerically simulating quantum computing for a decade. We introduce our C++ library ZKCM_QC developed for multiprecision TDMPS simulations of quantum circuits. Besides its practical usability, the library is useful for evaluation of the method itself. With the library, we can capture two types of numerical errors in the TDMPS simulations: one due to rounding errors caused by the shortage in mantissa portions of floating-point numbers; the other due to truncations of nonnegligible Schmidt coefficients and their corresponding Schmidt vectors. We numerically analyze these errors in TDMPS simulations of quantum algorithms.

1. Introduction

Simulations of time-evolving quantum states using a matrix-product-state (MPS) [1] representation have been widely used in a variety of physical systems [2, 3]. We have been developing a C++ library, named ZKCM_QC [4], for multiprecision time-dependent MPS (TDMPS) simulations of quantum computing using Vidal's representation [5] for MPS. Here, we define the precision by the length of a mantissa portion for each floating point number.

Let us begin with a brief description of conventions. We employ the computational basis $\{|0\rangle,|1\rangle\}^n$ for n-qubit quantum states, where $|0\rangle = \begin{pmatrix} 1 & 0 \end{pmatrix}^{\rm T}$ and $|1\rangle = \begin{pmatrix} 0 & 1 \end{pmatrix}^{\rm T}$. An n-qubit quantum state is represented as $|\Psi\rangle = \sum_{i_0\cdots i_{n-1}=0\cdots 0}^{1\cdots 1} c_{i_0\cdots i_{n-1}}|i_0\cdots i_{n-1}\rangle$ with complex amplitudes $c_{i_0\cdots i_{n-1}}$. In this paper, we employ the following MPS form [5, 6] of the state, for our TDMPS simulations.

$$\begin{aligned}|\Psi\rangle = \sum_{i_0\cdots i_{n-1}=0\cdots 0}^{1\cdots 1}\Bigg[&\sum_{v_0=0}^{m_0-1}\sum_{v_1=0}^{m_1-1}\cdots\sum_{v_{n-2}=0}^{m_{n-2}-1} Q_0(i_0,v_0)V_0(v_0)Q_1(i_1,v_0,v_1)V_1(v_1)\cdots \\ &Q_s(i_s,v_{s-1},v_s)V_s(v_s)\cdots V_{n-2}(v_{n-2})Q_{n-1}(i_{n-1},v_{n-2})\Bigg]|i_0\cdots i_{n-1}\rangle,\end{aligned} \tag{1}$$

where we use tensors $\{Q_s\}_{s=0}^{n-1}$ with parameters i_s, v_{s-1}, v_s (v_{-1} and v_{n-1} are excluded) and $\{V_s\}_{s=0}^{n-2}$ with parameter v_s; m_s is a suitable number of values for v_s, with which the state is represented precisely or well approximated. [Tensor $V_s(v_s)$ stores the Schmidt coefficients for the splitting between the sth site and the $(s+1)$th site.]

With the MPS form, the cost to update data in accordance with a time evolution is reduced considerably in comparison to the brute-force method. We have only to update tensors corresponding to the sites under the influence of a unitary operator for each step. The cost to simulate an evolution by a single unitary operation $\in$ U(4) acting on some consecutive sites is $O(m_{\max}^3)$ floating-point operations where $m_{\max}$ is the largest value of m_s among the sites s [5]. The total cost of a TDMPS simulation thus grows polynomially in the largest Schmidt rank among those for the splittings, which highly depends on the instance of the problem of one's concern.

The TDMPS simulation method in double precision [*i.e.*, (52+1)-bits-long mantissa for a floating point number] has already been used commonly in the community of computational physics [7]. There are, however, known cases for general computational methods where more accurate computation is needed to obtain a reliable results describing physical phenomena [8]. Accumulations of rounding errors of basic arithmetic operations are the main cause in such cases. This should be true also in TDMPS simulations of quantum computing where many matrix diagonalizations are involved unless the depth of quantum circuits is very small. Besides the precision of basic operations, another factor of losing accuracy is the truncation of Schmidt coefficients. When many of Schmidt coefficients are nonnegligible at a certain step of a TDMPS simulation, imposing a threshold to the number of Schmidt coefficients for each splitting may truncate out important data affecting simulation results [9]. So far truncations of nonvanishing Schmidt coefficients have been uncommon and the largest Schmidt rank and its upper bound in the absence of truncations have been of main concern when MPS and related data structures are used for handling quantum and/or classical computational problems [5, 10, 11, 6, 12, 13, 14, 15, 16]. This is in contrast to MPS and TDMPS simulations in condensed matter physics where truncations are very commonly employed [17].

In this report, we evaluate numerical errors in actual TDMPS simulations of quantum algorithms. We first begin with a brief introduction to the ZKCM_QC library in section 2. Then we conduct numerical evaluations in section 3: an error due to precision shortage is investigated in section 3.1 and that due to truncations of Schmidt coefficients is investigated in section 3.2. We discuss and summarize obtained results in sections 4 and 5, respectively.

2. ZKCM_QC library

The ZKCM_QC library has been developed as a C++ library for TDMPS simulations of quantum computing with an emphasis on multiprecision computation. It is an extension package of the ZKCM library [18] which uses the GMP [19] and MPFR [20] libraries for basic arithmetic operations. We briefly explain the usage of the ZKCM_QC library.

The library can be installed by the standard process: "`./configure`", "`make`", and "`make install`" in any Unix-like system with GNU tools. Once it is installed, a user will write a C++ program using the header file "`zkcm_qc.hpp`" and compile the program with the library flags "`-lzkcm_qc -lzkcm -lmpfr -lgmp -lgmpxx -lm`" using a C++ compiler.

In the main part of a program, a user will firstly specify the default precision. For example "`zkcm_set_default_prec(256);`" will set the default precision to 256 bits. Then, a user will call a constructor of the class "`mps`" to create the object of an MPS kept in the form of (1). For example, "`mps M(5);`" will generate an object keeping the data of $|0_0 0_1 0_2 0_3 0_4\rangle$ in the MPS form. Then a user will apply certain unitary operations by using some member functions of the class, such as "`applyU`" and "`applyU8`" with predefined and/or user-defined unitary matrices. For example, "`M.applyU(tensor2tools::Hadamard, 2);`" applies an Hadamard gate $H = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}/\sqrt{2}$ to qubit 2. For another example, "`M.applyU(tensor2tools::CNOT, 2, 4);`" applies the CNOT gate to qubits 2 and 4 that are the control bit and the target bit, respectively. Here, CNOT is a conditional bit flip; the target bit is flipped when the control

bit is in $|1\rangle$. There are other member functions for unitary operations. For instance, "`CCNOT`" is used for a CCNOT operation; "`M.CCNOT(0, 2, 4);`" applies a bit flip on qubit 4 under the condition that qubits 0 and 2 are both in $|1\rangle$. In addition, it is possible to simulate a projective measurement by calling the member function "`pmeasure`". For example, "`int event = M.pmeasure(zkcm_matrix("[1,0;0,-1]"), 2, -1);`" will simulate a projective measurement on qubit 2 with the observable Pauli Z, as a random process. The returned value "`event`" will be 0 when the event corresponding to the larger eigenvalue of Z occurs and will be 1 otherwise. There are other useful functions documented in the reference manual of the library. Among them, functions to achieve a reduced density matrix of specified qubits will be frequently used. One example to use such a function is "`std::string s = M.RDO_block(2, 4).str_dirac_b();`" which obtains the reduced density matrix of the block of qubits 2, 3, and 4 as a "`std::string`"-type string.

For each call of unitary operations and/or projective measurements, we update involved tensors of the MPS. This is a tedious process, but is concealed by the library. User programs usually do not pay attention to the background simulation process. One may still set a threshold m_{trunc} to the number of Schmidt coefficients of a bipartite splitting at one's risk. A user will write, *e.g.*, "`M.m_trunc(12);`" to specify the threshold (it is set to 12 for this example, *i.e.*, only largest 12 Schmidt coefficients will be kept at each time tensors are updated during the TDMPS simulation).

More detailed instructions on the installation and the usage of the ZKCM_QC library are found in the documents placed at the "`doc`" directory of the package.

3. Numerical errors in TDMPS simulations of quantum computing

In this section, we firstly investigate the influence of precision shortage in a TDMPS simulation of a simple quantum search [21] and secondly investigate the influence of the truncation of Schmidt coefficients in a TDMPS simulation of the Deutsch-Jozsa algorithm [22] for a simple boolean function.

3.1. Errors due to the precision shortage

A TDMPS simulation is in fact sensitive to the precision of floating-point computation. Here, we show a typical example.

We consider Grover's quantum search [21] (or quantum amplitude amplification) for a simple oracle[1] with a three-qubit input. The initial state is set to $|s\rangle = \frac{1}{\sqrt{8}} \sum_{x_0x_1x_2=000}^{111} |x_0x_1x_2\rangle$ and the target state is set to $|101\rangle$. We successively apply the so-called Grover routine[2] $R = (1 - 2|s\rangle\langle s|)(1 - 2|101\rangle\langle 101|)$ to the state starting from $|s\rangle$. This results in an oscillation of the population of $|101\rangle$, $\mathrm{Prob}_{101}(t) = |\langle 101|R^t|s\rangle|^2$, as shown in figure 1 ($t = 0, 1, 2, \ldots$). For the TDMPS simulation, we used five qubits including a single oracle qubit and a single ancilla qubit. The oracle circuit was simply constructed by three CCNOT gates and two NOT gates. This is just for simplified demonstration; a meaningful quantum search should be performed for a realistic computational problem like a satisfiability problem [23] with a large input size. This will be hopefully realized in future with a real quantum computer. The quantum circuit of the total process of our present concern is quite simple as illustrated in figure 2. In addition, here, truncations of nonzero Schmidt coefficients were not employed.

[1] A TDMPS simulation of the Grover search for a simple oracle was first performed by Kawaguchi *et al.* [10] in 2004.

[2] It is in general $(1-2|s\rangle\langle s|)(1-2\sum_{x\in X}|x\rangle\langle x|)$ with $|s\rangle$ the equally-weighted superposition of states in a parent set and X the set of target states. It is highly dependent on a problem instance how this operation is constructed as a quantum circuit.

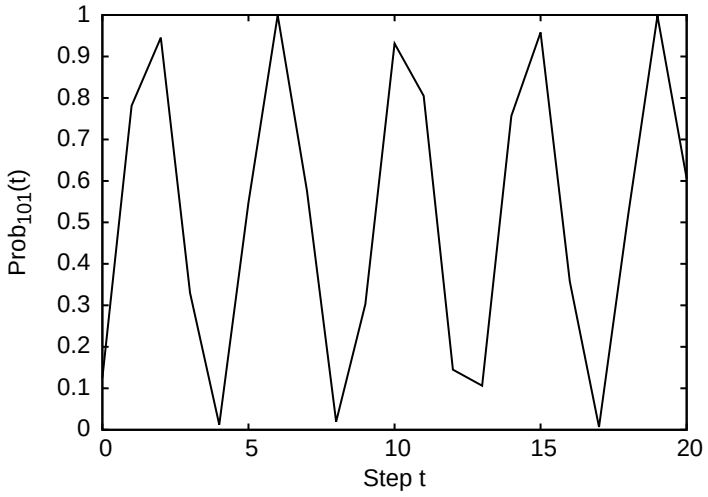

Figure 1. Theoretical graph of the oscillation of $\mathrm{Prob}_{101}(t)$ in the three-qubit quantum search described in the text.

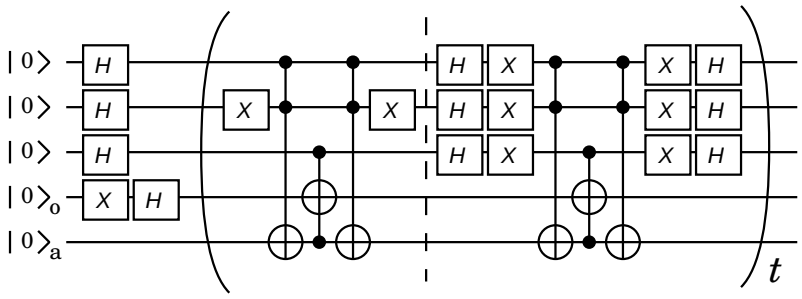

Figure 2. Quantum circuit of the three-qubit quantum search described in the text. The circuit in the parentheses illustrates R, in which the left half corresponds to $(1-2|101\rangle\langle 101|)$ and the right half corresponds to $(1-2|s\rangle\langle s|)$. Symbol "o" stands for the oracle qubit and "a" stands for the ancilla qubit.

We plot the computational error $|\widetilde{\mathrm{Prob}}_{101}(20) - \mathrm{Prob}_{101}(20)|$ after 20 iterations of the Grover routine against the precision in figure 3, where $\widetilde{\mathrm{Prob}}_{101}(20)$ is a computed value and $\mathrm{Prob}_{101}(20) = |\langle 101|R^{20}|s\rangle|^2 = \frac{5327874951961}{8796093022208}$ is the exact value calculated symbolically by the Maxima system [24] with the Qcomp.mac package [25]. It is manifest that the double precision (53 bits) is not enough and more than 70 bits precision is preferable for an accurate simulation. In addition, it is interesting that the computational error does not look like a simple elementary function of the precision; there is an abrupt drop at a certain value of the precision.

We next evaluate the computational cost by increasing the precision of the TDMPS simulation. The real CPU time used for simulating the above Grover search is plotted against the precision in figure 4. In the figure, we employed a polynomial for coarse fitting, since the computational cost of a TDMPS simulation of a quantum algorithm is guaranteed to be some polynomial in the floating-point precision p (in bits) because each basic arithmetic operation is performed within polynomial time. It can be exponential in the input size of a problem instance, but this is not of our present interest. One might expect that the cost grows quasi-linearly in p considering the cost of each floating-point multiplication. However, the value of p does effect on

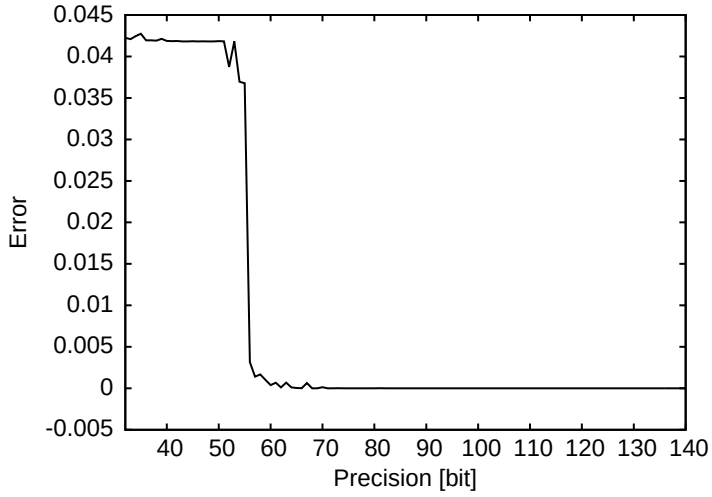

Figure 3. Computational error $|\widetilde{\mathrm{Prob}}_{101}(20) - \mathrm{Prob}_{101}(20)|$ plotted against the precision.

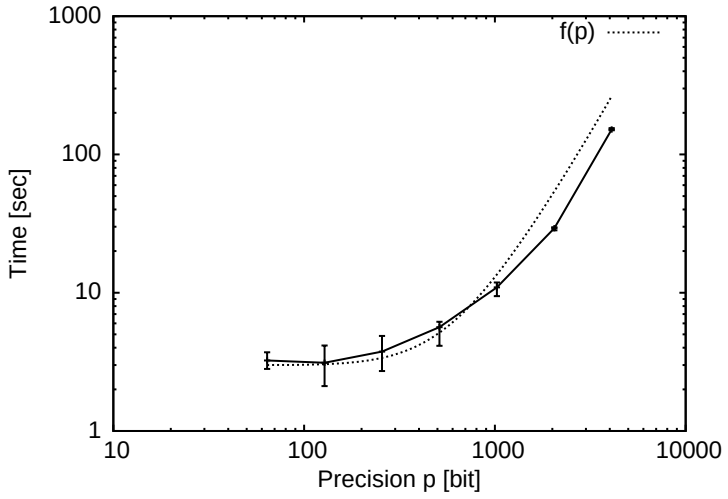

Figure 4. Computation time to simulate 20 iterations of the Grover routine, plotted against the precision. The average was taken over 10 trials for each data point. Error bars span from the minimum to the maximum. Here, $f(p) = 1.5 \times 10^{-9}(p-64)^3 + 1.0 \times 10^{-5}(p-64)^2 + 3.0$. Environment: Redhat Enterprise Linux 6 on Intel Xeon X7542 CPU 2.67GHz, 132GB memory.

the number of inverse iterations to guarantee enough accuracy in each matrix diagonalization. In this sense, the fitting seems plausible. The maximum Schmidt rank of the MPS at the time steps between basic quantum gates was 4 during the above simulation. One may find it smaller than expected. This is because operations $\in$ U(8) are simulated without decomposition in terms of U(4) and U(2) operations, in ZKCM_QC.

3.2. Errors due to truncations

It is uncommon to make use of a truncation in TDMPS simulations of quantum computing unlike those used for condensed matter physics as we mentioned in section 1. We however investigate the case we dare to impose truncations. Let us denote the limit to the number of Schmidt

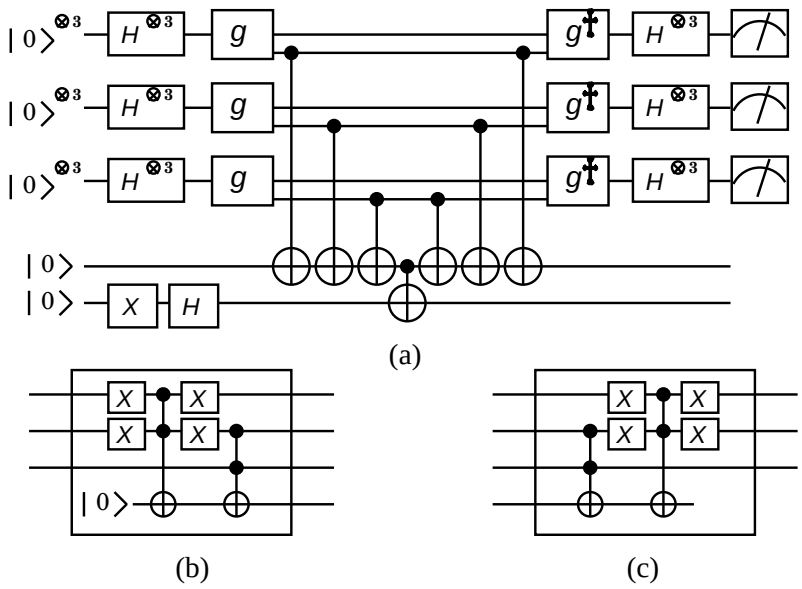

Figure 5. Quantum circuit of the Deutsch-Jozsa algorithm for the function f. Circuits (b) and (c) describe the gates g and $g^\dagger$ of circuit (a), respectively.

coefficients that can be kept for each splitting as $m_{\rm trunc}$. Thus, among Schmidt coefficients, larger $m_{\rm trunc}$ coefficients are kept and the remaining small coefficients are truncated out at each step in the simulation of a quantum circuit.

Here we study the effect of a truncation in a TDMPS simulation of the Deutsch-Jozsa algorithm [22] for a function $f : \{0,1\}^9 \to \{0,1\}$ with a nine-qubit input $\mathbf{x}$. Recall that f is either balanced (*i.e.*, $\sharp\{\mathbf{x}|f(\mathbf{x}) = 0\} = \sharp\{\mathbf{x}|f(\mathbf{x}) = 1\}$) or constant (*i.e.*, $f(\mathbf{x})$ is same for all $\mathbf{x}$) by the promise of the problem definition for the algorithm (see, *e.g.*, section 3.1.2 of [26]). The Deutsch-Jozsa algorithm for nine qubits is interpreted as the following process. (i) Apply $H^{\otimes 9}V_f H^{\otimes 9}$ to $|0_0 \dots 0_8\rangle$ (here, V_f is an operation to put the factor $(-1)^{f(\mathbf{x})}$ to the states $|\mathbf{x}\rangle$); (ii) Measure the nine qubits of the resultant state. When f is balanced, the probability of having the nine qubits in $|0\rangle$'s simultaneously in the resultant state vanishes; when f is constant, the probability is exactly unity.

We employ the following function for our TDMPS simulation. $f(\mathbf{x}) = g(x_0x_1x_2) \oplus g(x_3x_4x_5) \oplus g(x_6x_7x_8)$ with $g(\mathbf{y}) = [(\neg y_0) \wedge (\neg y_1)] \vee (y_1 \wedge y_2)$. This function is balanced. In addition, more specifically, we have $\mathrm{Prob}(0_0 0_1 0_2) = 0$ in the resultant state of (i) for this particular function. The quantum circuit of the Deutsch-Jozsa algorithm for this function is illustrated in figure 5. We set the floating-point precision to 256 bits and tried several different values of $m_{\rm trunc}$ in the TDMPS simulation of the quantum circuit. As shown in figure 6, the error in the computed value of $\mathrm{Prob}(0_0 0_1 0_2)$ (namely, a discrepancy from zero in the present context) vanishes for $m_{\rm trunc} \geq 12$ while a considerable error exists for $m_{\rm trunc} \leq 11$. The largest value among m_s's during the simulation is shown as $m_{\rm max}$ in the figure. It indicates that $m_{\rm trunc}$ should be set to at least the maximum possible value of $m_{\rm max}$ which is 12 in the present case, so as to avoid a nonnegligible error.

The above result shows that we cannot truncate out any nonzero Schmidt coefficient during the simulation. This phenomenon should be related to the distribution of Schmidt coefficients if discussions of reference [9] apply to the present case. We show the distribution at the points we had the maximum Schmidt rank 12 in the simulation, in figure 7 (this is same for both of the points). These points were the second CNOT gate from the left and one from the right among

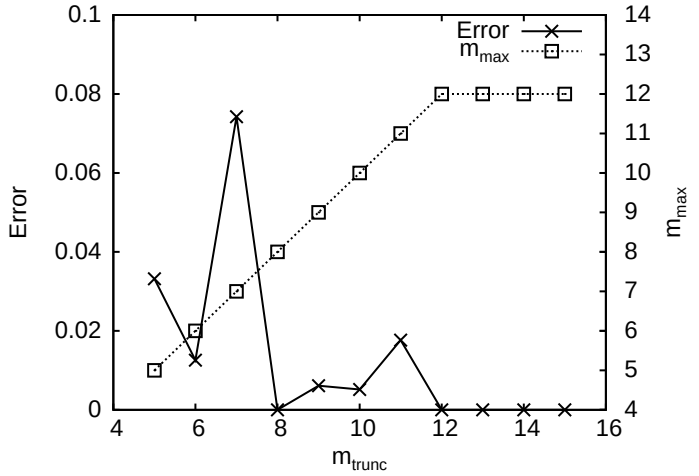

Figure 6. Plot of the error in the computed value of $\mathrm{Prob}(0_0 0_1 1_0 2_0)$ (crosses) and the plot of $m_{\max}$ (boxes) as functions of m_{trunc}. See the text for the details of the simulation.

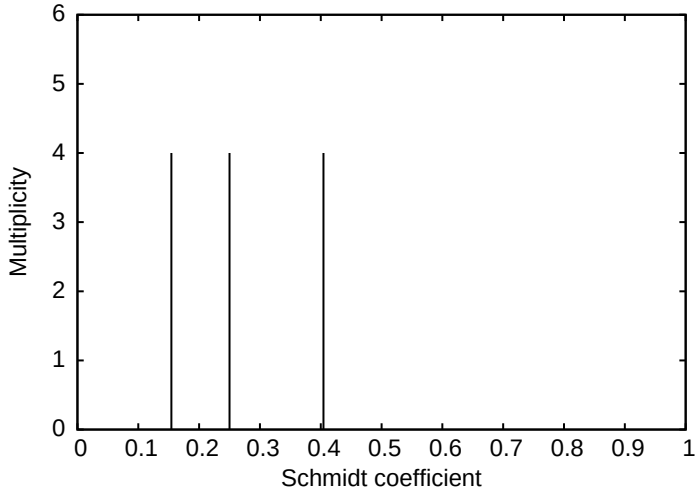

Figure 7. Distribution of Schmidt coefficients for the splitting at the points we had the maximum Schmidt rank in the simulation. The distribution was same for both of the points.

the seven CNOT gates in the middle part of circuit (a) of figure 5. It is now manifest that none of the twelve Schmidt coefficients are negligible. This is how even a single truncation caused a significant error.

4. Discussion

The TDMPS method is regarded as an approximation method in condensed matter physics, but is usually an exact method in simulation studies of quantum computing. Truncations of nonzero small Schmidt coefficients are thus unemployed, usually, for the purpose of simulating quantum computation. In a known case in condensed matter physics [9] where we should avoid truncations, a dominant number of Schmidt coefficients are nonnegligible. This should be true also in TDMPS simulations of quantum circuits. In fact, we found that this is the case and

the threshold m_{trunc} for the largest number of surviving Schmidt coefficients should be at least the maximum possible number of nonzero Schmidt coefficients, in the simulation results for the Deutsch-Jozsa algorithm presented in section 3.2.

Besides, we have also evaluated the effect of precision shortage. Rounding errors of individual basic arithmetic operations may accumulate to a large error in the resultant state of a TDMPS simulation of a quantum circuit. In our simulation of a simple quantum search presented in section 3.1, a nonnegligible error was observed in the resultant population of a target state unless we increase the precision considerably larger than the double precision. Furthermore, we found that the error does not behave like a smooth curve as a function of the precision in bits, but behaves like a step function with a sudden drop, as shown in figure 3. An error with such a behavior cannot be easily captured unless we go beyond the double precision. This is not a particular phenomenon for a TDMPS simulation but a rather commonly observed one for matrix computation (see the document of the ZKCM library [18]).

With our simulation results for investigating numerical errors, it is suggested that the precision should be at least 70 bits and the truncation of nonzero Schmidt coefficients is not encouraged for a TDMPS simulation of quantum computation. As for the cost of multiprecision computation, figure 4 indicates that more than 1000 bits precision is quite expensive. As a matter of fact, the current computer architecture does not have a good hardware support for high precision computation. Thus the practical precision we may employ should be between 70 bits and several hundreds bits for the time being.

5. Conclusion

We have utilized our multiprecision TDMPS library ZKCM_QC to investigate numerical errors in TDMPS simulations of quantum algorithms. To avoid a nonnegligible rounding error within a practical cost, it is suggested that the length of the mantissa portion of each floating-point number should be at least 70 bits but not more than 1000 bits in the present technology. It is also suggested that a truncation of nonzero Schmidt coefficients is discouraged in case of simulating quantum computation.

6. Software information

The simulations were performed with ZKCM_QC ver. 0.0.9beta put on the repository `https://sourceforge.net/p/zkcm/sublibqc` .

References

[1] White S 1993 *Phys. Rev. B* **48** 10345
[2] Schollwöck U 2005 *Rev. Mod. Phys.* **77** 259–315
[3] Schollwöck U 2011 *Ann. Phys.* **326** 96–192
[4] SaiToh A ZKCM_QC https://sourceforge.net/p/zkcm/home/Extension Packages/; see also *Preprint* arXiv:1111.3124
[5] Vidal G 2003 *Phys. Rev. Lett.* **91** 147902
[6] SaiToh A and Kitagawa M 2006 *Phys. Rev. A* **73** 062332
[7] Bauer B *et al.* 2011 *J. Stat. Mech.* **2011(05)** P05001; http://alps.comp-phys.org/
[8] Bailey D, Barrio R and Borwein J 2012 *Appl. Math. Comput.* **218** 10106–10121
[9] Venzl H, Daley A, Mintert F and Buchleitner A 2009 *Phys. Rev. E* **79** 056223
[10] Kawaguchi A, Shimizu K, Tokura Y and Imoto N Classical simulation of quantum algorithms using the tensor product representation *Preprint* arXiv:quant-ph/0411205
[11] Markov I and Shi Y 2008 *SIAM J. Comput.* **38** 963–981
[12] Yoran N and Short A 2006 *Phys. Rev. Lett.* **96** 170503
[13] Jozsa R On the simulation of quantum circuits *Preprint* arXiv:quant-ph/0603163
[14] Chamon C and Mucciolo E 2012 *Phys. Rev. Lett.* **109** 030503
[15] Temme K and Wocjan P Efficient computation of the permanent of block factorizable matrices *Preprint* arXiv:1208.6589

[16] Johnson T, Biamonte J, Clark S and Jaksch D Solving search problems by strongly simulating quantum circuits *Preprint* arXiv:1209.6010
[17] Hallberg K 2006 *Adv. Phys.* **55** 477–526
[18] Documents linked from http://zkcm.sf.net/
[19] The GNU multiple precision arithmetic library http://gmplib.org/
[20] Fousse L, Hanrot G, Lefèvre V, Pélissier P and Zimmermann P 2007 *ACM Trans. Math. Software* **33** 13; http://www.mpfr.org/
[21] Grover L K 1996 *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC 1996)* (Philadelphia, PA: ACM Press, New York, 1996) pp 212–219
[22] Deutsch D and Jozsa R 1992 *Proc. Royal Soc. London A* **439** 553–558
[23] Garey M and Johnson D 1979 *Computers and Intractability: A Guide to the Theory of NP-Completeness* (New York: W.H. Freeman and Co.)
[24] Maxima, a computer algebra system http://maxima.sourceforge.net/
[25] SaiToh A Qcomp.mac http://silqcs.org/~saitoh/qcomp.mac
[26] Gruska J 1999 *Quantum Computing* (Berkshire, UK: McGraw-Hill)